PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-289329
(43)Date of publication of application : 19.10.1999

(51)Int.Cl. H04L 9/32
G09C 1/00
G09C 1/00

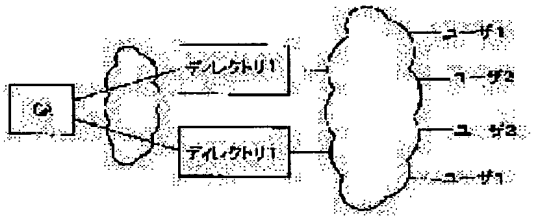
(21)Application number : 11-053084
(22)Date of filing : 22.01.1999

(71)Applicant : YEDA RES & DEV CO LTD
(72)Inventor : NAOR MONI
NISSIM YAACOV

(30)Priority
Priority number : 98 10571 Priority date : 22.01.1998 Priority country : US

(54) VERIFICATION TYPE SEARCH TREE

(57)Abstract:
PROBLEM TO BE SOLVED: To reduce wide calculation and communication cost by allowing each node value to apply an encryption hash function at least to slave node value and dynamic search value of the node and providing a root node which is authenticated with a search tree that comes into existence and digital signature.
SOLUTION: A certification authority(CA) updates a search tree, calculates an authentication path induced by an updated node, performs digital signature authentication of a node that is subjected to root correction and sends a corrected parameter to a directory 1. The directory 1 verifies if recalculated root value matches with root value from the CA. A user inquires of the directory 1 about a certificate continuous number, the directory 1 calculates an induced authentication path and transmits it to the user and the user verifies the item. For instance, when the directory 1 asserts that the certificate has been, the user applies a hash function and checks from a leaf to a root.



LEGAL STATUS
[Date of request for examination]
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平11－289329

(43)公開日　平成11年(1999)10月19日

(51)Int.Cl.[6]		識別記号	FI		
Ｈ０４Ｌ	9/32		Ｈ０４Ｌ	9/00	６７５Ｂ
Ｇ０９Ｃ	1/00	６４０	Ｇ０９Ｃ	1/00	６４０Ｂ
		６６０			６６０Ａ

審査請求　未請求　請求項の数９　ＯＬ　外国語出願　（全 44 頁）

(21)出願番号　特願平11－53084

(22)出願日　平成11年(1999)１月22日

(31)優先権主張番号　０１０５７１

(32)優先日　1998年１月22日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　591015175
イエダ　リサーチ　アンド　デベロップメント　カンパニー　リミテッド
イスラエル国レホボト　ピー　オー　ボックス　95

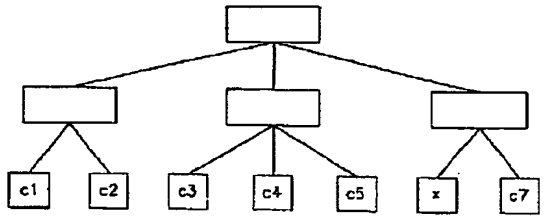
(72)発明者　モニ，ナオル
イスラエル国テルアビブ，ベイト　－　ソリ　ストリート　5

(72)発明者　ヤーコブ，ニッシム
イスラエル国ラマト　－　ガン，ハルジム　ストリート　28

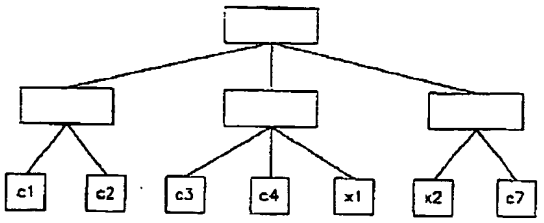
(74)代理人　弁理士　浅村　皓　（外３名）

(54)【発明の名称】　認証形サーチ・ツリー

(57)【要約】
【課題】　集合内のアイテムのメンバーシップまたは非メンバーシップを認証するよう作用する認証形サーチ・ツリーを提供する。
【解決手段】　認証形サーチ・ツリーは、ノードとリーフを有しかつこれらに関連してサーチ手法を有したサーチ・ツリーを含む。ノードは、ダイナミック・サーチ値を含み、リーフは集合のアイテムを含む。ノードの各々には、暗号化ハッシュ関数値を関連付け、この暗号化ハッシュ関数値は、子ノードの暗号化ハッシュ値と、ノードのダイナミック・サーチ値とに暗号化ハッシュ関数を適用することにより発生する。認証形サーチ・ツリーのルート・ノードは　デジタル署名により認証する。



A



B

【特許請求の範囲】
【請求項1】　集合内のアイテムのメンバーシップまたは非メンバーシップを認証するよう作用する認証形サーチ・ツリーを含むメモリにおいて、前記認証形サーチ・ツリーが、
ノードとリーフを有しかつこれらに関連してサーチ手法を有したサーチ・ツリーであって、前記ノードがダイナミック・サーチ値を含み、前記リーフが前記集合のアイテムを含み、前記ノードの各々には、暗号化ハッシュ関数値を関連付け、該暗号化ハッシュ関数値は、少なくとも（Ｉ）子ノードの暗号化ハッシュ値と、（II）前記ノードの前記ダイナミック・サーチ値と、に暗号化ハッシュ関数を適用することにより発生する、前記のサーチ・ツリーと、
デジタル署名により認証した前記認証形サーチ・ツリーの少なくともルート・ノードと、を備えたこと、を特徴とする認証形サーチ・ツリー。
【請求項2】　請求項1記載のサーチ認証形ツリーにおいて、前記暗号化ハッシュ関数がユニバーサル一方向性関数タイプのものであり、前記暗号化一方向性関数を、各内部ノードに固有のユニバーサル一方向性関数にさらに適用したこと、を特徴とするサーチ認証形ツリー。
【請求項3】　請求項1記載のサーチ認証形ツリーにおいて、前記サーチ・ツリーはBtreeであること、を特徴とするサーチ認証形ツリー。
【請求項4】　請求項1記載のサーチ認証形ツリーにおいて、前記サーチ・ツリーは2－3ツリーであること、を特徴とするサーチ認証形ツリー。
【請求項5】　集合内のアイテムのメンバーシップまたは非メンバーシップを認証する方法が、
（ｉ）請求項1記載の認証形サーチ・ツリーを提供し、
（ii）前記集合の少なくとも1つのアイテムの認証を、該少なくとも1つのアイテムと前記ルートにより誘起される認証パスを計算することにより行うこと、から成る認証方法。
【請求項6】　集合内の少なくとも1つのアイテムを認証形サーチ・ツリーにおいて更新する方法が、
（ｉ）請求項1記載の認証形サーチ・ツリーを提供し、
（ii）前記サーチ・ツリーを更新することにより更新したノードを獲得し、
（iii）前記更新したノードにより誘起された認証パスを計算し、
（iv）少なくとも前記のルート修正したノードをデジタル署名により認証すること、から成る更新方法。
【請求項7】　請求項5記載の方法において、ＣＡ、ディレクトリ、ユーザの手法において、前記ステップ（ii）は、
（ａ）前記ユーザが、集合内の少なくとも1つのアイテムのメンバーシップまたは非メンバーシップを認証するため、前記少なくとも1つのアイテムのリストをディレクトリに提供し、
（ｂ）前記ディレクトリが、前記少なくとも1つのアイテムにより誘起された前記認証パス（複数）を計算しこれをユーザに送信し、前記ディレクトリがさらに前記認証パスを送信し、
（ｃ）前記ユーザが前記アイテムを検証すること、を含むこと、を特徴とする認証方法。
【請求項8】　請求項6記載の方法において、ＣＡ、ディレクトリ、ユーザの手法において、
前記ＣＡが
（ｉ）前記サーチ・ツリーを更新することにより、更新したノードを獲得し、
（ii）前記更新したノードにより誘起された認証パスを計算し、
（iii）少なくとも前記のルート修正したノードをデジタル署名により認証し、
（iv）修正したパラメータを前記ディレクトリに送信すること、を実行し、
前記ディレクトリが、
（ｉ）前記修正パラメータを適用することにより認証したディレクトリ・ルート値を獲得し、
（ii）前記認証したＣＡルート値が前記認証したディレクトリ値と一致したか検証すること、を実行すること、を特徴とする認証方法。
【請求項9】　請求項6記載の方法において、ＣＡ、ユーザの手法においては、
前記ＣＡが、
（ｉ）前記サーチ・ツリーを更新することにより更新したノードを獲得し、
（ii）前記更新したノードにより誘起された認証パスを計算し、
（iii）少なくとも前記のルート修正したノードをデジタル署名により認証し、
（ｖ）誘起されたサブツリーを前記ユーザに送信すること、を実行し、
前記ユーザが、
（iii）前記誘起されたサブツリーをユーザの自己パスと交差させそしてユーザ認証されたルート値を獲得し、
（iv）前記認証されたＣＡルート値が前記認証されたユーザ値と一致したか検証すること、を実行すること、を特徴とする認証方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、認証目的のためのデジタル署名の一般的な分野にある。
【０００２】
【従来の技術】公開鍵暗号の幅広い使用は、公開鍵の真正さを検証する能力を必要としている。これは、“公開鍵インフラストラクチャ（Public Key Infrastructure (PKI)）”における証明書の使用を通して実現される。

証明書とは、公的信用機関（証明機関（ＣＡ：certification authority）であって、これの公開鍵の真正さは他の手段によって提供され得る機関）により署名されたメッセージであり、これは、公開鍵と、追加の情報、例えば、満了日、連続番号、および鍵と対象エンティティに関する情報を含んでいる。

【０００３】証明書が発行されたとき、その有効性は、満了日によって制限されている。しかし、証明書をその満了日前に取り消さなければならない状況がある（例えば、秘密鍵が漏れたとき、あるいは鍵保有者が加入または地位を変更したとき）。したがって、証明書が存在することは、必要なことであるが、その有効性に対する十分な証拠となるものではなく、そのため証明書が取り消されたかどうかについて判定する機構が必要である。

【０００４】応用の代表的なものは、クレジットカード・システムであり、このシステムにおいては、クレジット会社が、クレジットカードを、例えばこのカードが盗難にあったと報告があったとき、あるいはそのユーザの銀行口座の残高にしたがって、その満了前に一時的にあるいは恒久的に取り消すことができる。

【０００５】

【発明が解決しようとする課題】証明書取り消しリスト（Certificate Revocation List (CRL)）

ＣＲＬは、ＣＡの発行する署名されたリストであって、取り消された証明書全てをそれらの連続番号により識別したものである。このリストは、（その新しさを示すものとして）タイムスタンプと連結され、そしてそれら証明書を最初に発行したＣＡが署名する。これらＣＲＬは、たとえ変更がなくても、定期的にディレクトリに送って、新しいＣＲＬではなく古いＣＲＬの悪意のリプレイを防止する。

【０００６】ディレクトリは、照会に対する回答として、最も更新されたＣＲＬを供給する（完全なＣＲＬは商人に送られる）。

・この手法の主たる利点は、その簡単さにある。

・この手法の主たる欠点は、その高いディレクトリーユーザ間の通信コストである（その理由は、ＣＲＬは非常に長くなることがあるからである）。別の欠点は、ユーザが、その証明書の有効性に関する簡潔な証拠を持ち合わせていないことがあることである。

【０００７】証明書に対しては、合理的な有効性満了期間を選ぶべきである。この満了期間が短いと、証明書を再発行するのに資源を浪費する。満了期間が長いと、ＣＲＬが長くなることがあり、通信コスト並びにＣＲＬ管理の困難さを高くしてしまう。カウフマン外（Kaufman et al. [15, Section 7.7.3]）は、ＣＲＬがある限界を超えたときには常に全ての証明書を再発行することを提案していた。その提案においては、証明書は、満了日ではなく連続番号を付す。（連続番号は、発行した各証明書に対して増分する。連続番号は、証明書全てを再発行するときでも再使用しない。）。このＣＲＬは、“第１有効”証明書を示すフィールドを含んでいる。全ての証明書を再発行したとき、ＣＲＬの第１有効証明書フィールドを更新することにより、第１の再発行した証明書の連続番号を含むようにする。

【０００８】証明書取り消しシステム

ミカリ（Micali [18]）は、ＣＲＬ通信コストを改善するため、証明書取り消しシステム（ＣＲＳ）を提案している。その基礎となる思想は、どの証明書のメッセージにも署名して、これが取り消されたかどうかについて述べ、そしてオフライン／オンライン署名法 [11]を使用することによりこれら署名を周期的に更新するコストを低減することである。

【０００９】証明書を作るため、ＣＡは、各証明書に対し、２つの番号(Y_{365}, N)を関連付け、これらは、’伝統的’な証明書データと共に署名する。各証明書に対しては、ＣＡは、２つの番号N_0, Y_0を（擬似）ランダムに選び、そして（一方向性関数fを使って）$Y_{365}=f^{365}(Y_0)$および$N=f(N_0)$を計算する。（実際には、fに関するより強力な仮定を必要とする、例えばfはその反復が一方向性である、すなわち、$y=f^i(x)$が与えられたときに、$y=f(x')$となるようなx'をみつけることは実行不可能である。これは、fが一方向性転置である場合に自動的に保証される。

【００１０】ディレクトリは、毎日、各証明書に対して番号Ｃを送るＣＡによって、以下のように更新する。

１．非取り消しの証明書に対して、ＣＡは、fの１つのアプリケーション、すなわち、$C=Y_{365-i}=f^{365-i}(Y_0)$を明らかにし、これにおいて、iは毎日増分するカウンタであり、発行日においてはi＝０である。

２．取り消した証明書に対しては、$C=N_0$である。したがって、Ｃの最も更新された値は、短い証拠（この証明書xは取り消されたあるいは取り消されたものではない）として作用し、これは、ディレクトリが、ユーザの照会xに対する返答として提示することができる。

【００１１】・　ＣＲＬに優るＣＲＳの利点は、その照会通信コストにある。連邦ＰＫＩ（Public Key Infrastructure）評価に基づき、ミカリ（Micali [18]）は、ＣＲＳの毎日の更新は、ＣＲＬの更新よりはより費用がかかるが、ＣＲＳ照会のコストははるかに低い、ということを示した。彼は、ＣＲＬに対して通信コスト全体で９００倍の向上が生じると見積もった。

【００１２】ＣＲＳの別の利点は、各ユーザが、彼の証明書の有効性の簡潔な譲渡可能の証拠を保有することができる、ということである。ユーザがそのような証拠を保持していてしかも彼らの証明書と共に提示するときには、ディレクトリ・アクセスを省略する。

・　このシステムの主たる欠点は、ＣＡ－ディレクトリ間の通信の増大である（これは、ディレクトリーユーザ間の通信と同じ量であって、ディレクトリの存在がＣＡ

の通信を減少させると思われる。）。さらに、ＣＡの通信コストはディレクトリ更新レートに比例するため、ＣＡ－ディレクトリ間の通信コストは、ディレクトリの更新レートを制限する。

【００１３】証明書が取り消されていなかったということを検証することの複雑さは、更新レートにも比例する。例えば、１時間に１回の更新では、ユーザは、証明書が取り消されていなかったことを検証するために関数fを 365 x 24 = 8760回適用しなければならず、これは検証における主要なファクタとなる。

【００１４】証明書取り消しツリー

コーカー（Kocher〔16〕）は、証明書が取り消されていなかったことの短い証拠を証明書の検証者が得ることができるようにするため、証明書取り消しツリー（認証ツリーとも呼ぶ）の使用を提案している。ＣＲＴは、ハッシュ・ツリーであって、そのリーフがＣＡが発行する証明書連続番号Ｘについてのステートメント集合CA_Xに対応したものである。このステートメント集合は、各ＣＡの取り消した証明書の集合から発生する。これは、証明書Ｘを取り消したか否か（あるいはその状態がＣＲＴ発行者には未知であるかどうか）についての情報を提供する。２つのタイプのステートメントがある。すなわち、未知のＣＡのレンジを指定するものと、証明書レンジを指定するものであってこれの下側の証明書のみが取り消されているものと、である。例えば、CA_1が２つの証明書X1<; X2を取り消した場合、それらステートメントの内の１つは、以下の通りとなる。

【００１５】$CA_X = CA_1$および $X_1 \leq X <; X_2$の場合、X = vのときにXは取り消される。ＣＲＴを発生するには、ＣＲＴ発行者は、上記ステートメントに対応したリーフをもつ２進ハッシュ・ツリー〔17〕を構築する。

【００１６】証明書状態に関する証拠は、ハッシュ・ツリーにおけるルート（root）から、パス上の各ノードに対してその子供の値を指定する適切なリーフ（ステートメント）までのパスである。

・　ＣＲＬに優るＣＲＴの主な利点は、ＣＲＬ全体がある特定の証明書を検証するのに必要でないこと、そしてユーザが彼の証明書の有効性の簡潔な証拠を保持することができること、である。

・　ＣＲＴの主な欠点は、ＣＲＴの更新に計算作業が必要となることである。取り消した証明書の集合におけるいかなる変更も、ＣＲＴ全体の再計算を生じることになる。

【００１７】引用した従来技術のリスト

米国特許4,309,569（以下マークル（Merkle）特許と呼ぶ）

〔1〕 A. V. Aho, J. E. Hopcroft, J. D. Ullman. "Data Structures and Algorithms". Addison-Wesley, 1983.

〔2〕 R.G. Seidel., C.R Aragon "Randomized Search Trees". Proc. 30th Annual IEEE Symp. on Foundations of Computer Science, pp. 540-545, 1989.

〔6〕 M. Bellare, P. Rogaway. "Collision-Resistant Hashing: Towards MakingUOWHFs Practical". Advances in Cryptology - CRYPTO '97, Lecture Notesin Computer Science, Springer-Verlag, 1997.

〔11〕 S. Even, O. Goldreich, S. Micali. "On-Line/Off-Line Digital Signatures". Journal of Cryptology, Springer-Verlag, Vol. 9 pp. 35-67, 1996.

〔12〕 O. Goldreich, S. Goldwasser, and S. Halevi,. "Collision-Free Hashing from Lattice Problems". ECCC, TR96-042, 1996.
http://www. eccc. unitrier. de/eccc/

〔13〕 A. Herzberg, H. Yochai. "Mini-Pay: Charging per Click on the Web".Proc. 6th International World Wide Web Conference, 1997.
http: //www6. nttlabs. com/

〔15〕 C. Kaufman, R- Perlman, M. Speciner. "Network Security. Private Communication in a Public World". Prentice Hall series in networking anddistributing systems, 1995.

〔16〕 P. Koeher. "A Quick Introduction to Certificate Revocation Trees (CRTs)".
http://www. valicert. com/company/crt. html

〔17〕 P, C. Merkle. "A Certified Digital Signature". Proc. Crypto '89,Lecture Notes in Computer Science 435, pp. 234-246, Springer-Verlag, 1989.

〔18〕 S. Micali. "Efficient Certificate revocation". Technical Memo MIT/LCS/TM542b, 1996.

【００１８】用語解説

以下は、用語の解説であって、用語の一部は在来のものであり、また一部は新語である。

【００１９】証明機関（Certification Authority (CA)） － 信頼のおける者（trusted party）であって、すでに証明された公開鍵を有し、公開鍵および／またはクレジットカード番号のようなその他の情報の真正さを確立しその裏付けを行う責任がある者。

【００２０】ＣＡは、必ずしも必要ではないが、好ましくは、ユーザに対しオンライン証明書情報サービスを提供しない。その代わりに、ＣＡは、ディレクトリを定期的に更新する。以下に示すように、ある実施形態においては、ディレクトリを使用しない。

【００２１】ＣＡは、ユーザの証明書を、証明書の連続番号関連のデータと満了日とを含むメッセージにより発行する。この証明書は、ディレクトリへ送ることとユーザへ与えることとの一方または両方を行う。ＣＡは、証明書をその満了日前に取り消すことができる。証明書は、決して後者の定義に拘束されず、例えば１またはそれ以上（例えば、レンジ）の公開鍵、クレジットカード番号（単数または複数）、およびその他に関するデータ

を包含することができ、明示的な形態であるいはエンコーディングまたは暗号化のような関数を受けさせた後に提示する。（用語“アイテム”と“証明書”は、本明細書で交換可能に使用する。）

【００２２】ディレクトリ（Directory）　－　信頼のおけない１以上の者であって、ＣＡから更新された証明書取り消し情報を得て、そしてユーザがアクセス可能な証明書データベースとして作用する者。

【００２３】ユーザ（User）　－　信頼のおけない者であって、ＣＡからその証明書を受け、そして証明書情報に対する照会を発する者。ユーザは、とりわけ、以下を含むように構成すべきである。

(i) 他のユーザの証明書の有効性の照会を行う商人、(ii) 彼／彼女の証明書の有効性の証拠を、他のユーザと対面してそれを使用するために得るユーザ。

【００２４】サーチ・ツリー（Search tree）　－　ルートからサーチしているアイテム（リーフと関連する）までのツリー内のサーチ・パスを構成できるようにするサーチ手法に関連した周知のデータ構造。サーチ・パスは、ツリー・ノードそしておそらくはそのリンク内に存在するサーチ値を利用する。サーチ・ツリーは、更新取引を取り扱う（すなわち、アイテムをツリーから削除したりあるいはそれに追加したりする）ように本質的に設計されている。代表的には、排他的なものではないが、サーチ・ツリーの例は、2-3 tree, Btree, Btree+, TriS, treaps等である。

【００２５】更新取引（Update Transactions）　－　新たなアイテムをツリーに挿入し、ツリー内の既存のアイテムを削除する。

【００２６】認証ツリー（Authentication Tree）　－　ルートをもつツリーであって、各内部ノードが、その子供の値を暗号化ハッシュ関数によって認証し、そしてそのルートが、デジタル署名によって認証されたもの。代表的には、排他的なものではないが、例は、マークル特許に例示されている。

【００２７】暗号化ハッシュ関数（Cryptographic hash function）　－　これは以下を含む。

(i) 衝突イントラクタブル関数（collision intractable function）h()であって、h(x)= h(y)を満たす y≠xを見つけるのが計算上本質的に実行不可能であるようなもの。代表的には、排他的なものではないが、その例が、マークル特許に例示されている。または、(ii) ユニバーサル一方向性関数であって、関数h()のファミリが存在していて、そのファミリからのどのxおよびランダムなh()に対しても、h(x)= h(y)を満たす y≠xを見つけるのが計算上本質的に実行不可能であるもの（(I) および(II)におけるその詳細な説明については、[6]を参照）。

【００２８】

【課題を解決するための手段】したがって、当該分野においては、これまでに知られた技術に関連する欠点を、アイテムを認証する新規な技術を提供することにより、取り除くかあるいは実質的に削減する必要性がある。

【００２９】本発明は、在来の認証ツリー、並びに２－３ツリーまたはBtreeのような在来のサーチ・ツリーの利用を組み込む。サーチ・ツリーの利用は、アイテム（または複数のアイテム）の認証を可能にする一方、この目的のために大量のデータを送信する必要を取り除く。従来技術による認証ツリーの利用は、一連の取り消されたアイテムあるいは（他の場合には）有効なアイテムを送信できるようにする。

【００３０】認証ツリー、例えばマークル特許に開示された種類のものを使用するときの主要な欠点は、それが修正取引を受けるときに生ずる。これは、リーフ中のアイテムの新たな配列をもたらし、したがってその結果として、（以下に例示するように）ツリー中の多数のノード（以下、修正ノード）の値の修正を必要とする。

【００３１】修正ノードの値の更新のために広範な計算が必要となるだけでなく、認証を利用することにより、前記修正ノードの多数の値を例えばＣＡからディレクトリに通信ネットワークを介して送信するときに高い通信オーバーヘッドが課される。そのような修正がかなり頻繁に発生することがあることを考慮すると、その述べたオーバーヘッドは、従来技術の認証ツリーの使用を商業的に実行不可能にする。

【００３２】本発明によれば、在来の認証ツリーを、在来のサーチ・ツリーに“重畳”し（認証サーチ・ツリーをもたらし）、したがって、アイテムの認証に関する認証ツリーの固有の利点と、サーチ・ツリー構造によるツリー・ノードに課される制限された変更と、の両方から利益が得られる。

【００３３】したがって、本発明は、集合内のアイテムのメンバーシップまたは非メンバーシップを認証するよう作用する認証形サーチ・ツリーを含むメモリを提供し、前記認証形サーチ・ツリーが、ノードとリーフを有しかつこれらに関連してサーチ手法を有したサーチ・ツリーであって、前記ノードがダイナミック・サーチ値を含み、前記リーフが前記集合のアイテムを含み、前記ノードの各々には、暗号化ハッシュ関数値を関連付け、該暗号化ハッシュ関数値は、少なくとも（Ｉ）子ノードの暗号化ハッシュ値と、（II）前記ノードの前記ダイナミック・サーチ値と、に暗号化ハッシュ関数を適用することにより発生する、前記のサーチ・ツリーと、デジタル署名により認証した前記認証形サーチ・ツリーの少なくともルート・ノードと、を備える。さらに、本発明は、集合内のアイテムのメンバーシップまたは非メンバーシップを認証する方法を提供し、該方法が、（ｉ）指定した種類の認証形サーチ・ツリーを提供し、（ii）前記集合の少なくとも１つのアイテムの認証を、該少なくとも１つのアイテムと前記ルートにより誘起される認証パス

を計算することにより行うこと、から成る。さらに、本発明は、集合内の少なくとも１つのアイテムを認証形サーチ・ツリーにおいて更新する方法を提供し、該方法が、（ｉ）指定した種類の認証形サーチ・ツリーを提供し、（ii）前記サーチ・ツリーを更新することにより、更新したノードを獲得し、（iii）前記更新したノードにより誘起された認証パスを計算し、（iv）少なくとも前記のルート修正したノードをデジタル署名により認証すること、から成る。

【００３４】注意すべきであるが、上記の指定した順序は、反復的手順において全てのステップを各反復において実行する、ということを必ずしも意味するものではない。したがって、例えばステップ（ii）および（iii）は、各反復において実行し、そしてステップ（iv）は最後の反復において１回だけ適用することができる。これは、同様に、本文に記述する方法およびシステムの他のステップにも当てはまる。

【００３５】本発明は、さらに、必要な変更を加えた認証／更新のためのシステムを提供する。

【００３６】

【実施の形態】本発明について、図面を参照して詳細に説明する。

【００３７】まず図１を見ると、これは、従来技術、例えば特定したマークル特許に開示されたものによる認証ツリーを示している。尚、マークル特許の内容は、この参照により本開示に含めるものとする。

【００３８】例えば、証明書Y_1－Y_8が全ての有効なクレジットカードの証明書リスト（ＣＬ）に対して有効である場合について考察する。ここで、ユーザが、商人と対面しての商取引において、彼のクレジットカードY_5の使用を希望しているとする。商人は、図１に開示した種類の認証ツリー（すなわち、有効なクレジットカードY_1－Y_8に対する認証ツリー）を保有するディレクトリをアドレス指定する。ここで想起されるように、このディレクトリは、信頼のおけない者であり、したがって商人は、Y_5がこのツリー内に確かに現れているかについて検証をしたい。

【００３９】衝突イントラクタブル関数h()は、クレジットカード番号、例えば昇順にしたがって記憶したアイテム（クレジットカード）Y_1－Y_8を認証するために作用する。したがって、Y_5を認証するには、ディレクトリが商人に対し、ツリーのリーフとノードの値Y_5，H(6,6,Y)，H(7,8,Y)およびH(1,4,Y)を送信することで十分である。ただし、そのルート値H(1,8,Y)が、デジタル署名を使って、その前に認証されていたと仮定する。もちろん、追加のツリー値も送信することができるが、以下の説明から判るように、追加のツリー値を送信することは全く冗長なものである。

【００４０】したがって、Y_5を認証するには、商人（前のH()を知っている）は、認証パス、すなわち、（Y_5に基づいて）H(5,5,Y)、そしてH(5,5,Y)と受けたH(6,6,Y)とに基づいて、商人は、H(5,6,Y)を計算する。この後者は、受けたH(7,8,Y)と合わさってH(5,8,Y)を生じる。この後者は、受けたH(1,4,Y)と合わさってH(1,8,Y)を生じ、これは、ＰＫＩ技術を受けさせ（例えば、公開鍵ｎを適用する）、そしてその結果は、それ以前に認証したH(1,8,Y)値と比較し、そして一致した場合には、アイテムY_5はこの有効クレジットカードリストに属していることを保証する。

【００４１】もちろん、この認証ツリーの利点は、ほんのいくつかのノード値をユーザに送信し、そしてユーザがそれにも拘わらず、問題のアイテムY_5を認証することができることである。以下に説明するように、従来技術の認証ツリーに関してアイテムを認証するためのこの詳細な説明は、本発明のサーチ認証形ツリーにも同様に当てはまる。

【００４２】図１の認証ツリーの主要な欠点は、例えば新たなクレジットカードをＣＡにおけるリストに追加する場合に、この認証ツリーに修正取引を受けさせるときに生ずる。Y4 <; Y4' <; Y5となる新たなアイテムY4'を追加するとする。この結果の認証ツリー（図示せず）は、このツリー内のノードのほとんどについての広範な更新、並びに更新情報の過度の伝送オーバーヘッドを必要とすることになり、これは、明らかに望ましくなく、ＣＡを新たなアイテムで更新するレートが一般にかなり高いことを考慮すれば特にである。

【００４３】無効にしたまたは取り消したアイテム（例えば、無効のクレジットカード）を保有する証明書取り消しリスト（ＣＲＬ）を考慮するときには、上記の利点および欠点が等しく当てはまる。

【００４４】次に、本発明の１実施形態による例示のサーチ認証形ツリーについて、例えばＣＲＬ（例えば、リーフにおいて保持された取り消したクレジットカードのリスト：図２のＡ，Ｂを参照）を備えた２－３サーチ・ツリーを利用して考察する。

【００４５】これに関連して、注意されたいことに、本発明は、このサーチ・ツリーとこの目的に利用する任意の既知の技術との実際の実現法に決して限定するものではなく、特定の応用に依存して必要とされまた適当となる全ての場合に適用可能である。したがって、限定目的でない例として、アイテムを保持するどのような方法も適用可能であり、例えば、レコード、リンク・リスト、ツリー、ブロック（長いアイテムの場合）等である。この記述は、認証ツリーに対しても同様に有効である。

【００４６】したがって、この特定の実施形態により、２－３ツリーは、昇順での取り消した証明書の連続番号（ｃ１－ｃ７）に対応するリーフを備えて維持する。（２－３ツリーにおいては、あらゆる内部ノードは、２つまたは３つの子を有し、そしてルートからリーフへのパスは同じ長さを有する。）。メンバーシップをテスト

しそして修正する、すなわち単一のエレメントを挿入し、削除しあるいは更新することは、対数時間で行い、ここで、その修正は、その修正パス上のノードにのみ影響を与える。２－３ツリーの詳細な提示については、〔1, pp. 169-180〕を参照されたい。２－３ツリーの特性は、テストおよび修正がサーチ・パス上のノードに対する変更のみを含むこと、すなわち、あらゆる変更は、局部的であって、影響されるパスの数は少ない。

【００４７】このツリーは、最初は空の２－３ツリーに取り消した証明書の連続番号を１つ１つ挿入することにより作成するか、あるいは、連続番号のリストを分類しそしてこの分類したリスト内の連続番号に対応するリーフをもつ2段階ツリー（degree 2 tree）を構築することによって、作成することができる（理由は、通信の複雑さは、ツリーが２段階のものでは最小となる）。どのツリー・ノードにも、以下の手順による値を割り当てる。

・　各リーフは、取り消した証明書の連続番号をその値として記憶する。

・　内部ノードの値は、暗号化ハッシュ関数H()をその子供の値とそして内部ノード（これは、当てはまる場合には常に、リンクも包含する）の少なくともダイナミックなサーチ値とに適用することによって計算する。義務ではないが、暗号化一方向性ハッシュ関数H()は、ノードに関連したダイナミックなサーチ値以外の情報にも適用可能であり、例えば、ツリーをバランスさせるのに関連した情報等である。

【００４８】衝突イントラクタブル関数とは異なって、ユニバーサル一方向性ハッシュ関数を内部ノードに対し指定した方法で適用することは、各ノードに対する固有の関数の利用を必要とする。この場合、上記に加えて、子供の値と内部ノードのダイナミック・サーチ値と、そしてこの内部ノードに関連したこの関数の固有の値とを認証する必要がある。

【００４９】次に、以下は、本発明の１実施形態によるサーチ認証形ツリーの修正法に関する説明である。

【００５０】したがって、１つのアイテムを削除するには、在来の２－３アイテム削除ステップを実行する。すなわち、

１．　各々の満了した証明書の連続番号を２－３ツリーから削除し、削除パス上のノードの値を更新する。

【００５１】同様に、１つのアイテムを挿入するには、在来の２－３アイテム挿入ステップを実行する。すなわち、

２．　各々の新しい取り消した証明書連続番号をツリーに挿入し、この挿入パス上のノードの値を更新する。ツリー更新の間、２－３ツリーのバランスのため、新たなノードのあるものを作成したりあるいはあるノードを削除したりすることができる。これらノードは、挿入／削除したノード（以下、修正ノード）に対するサーチ・パス上においてのみ生起する。

【００５２】証明機関は、このツリーの認証を、そのルートの認証により行い、そしてこの目的のために、修正したノードにより誘起されたサーチ・パスのみを比較すべきである。

【００５３】このサーチ認証形ツリーのより簡単な実現のためには、２－３ツリーではなく他のツリー例えばランダム・トリープ（random treap）〔2〕を使用することができる。トリープは、２進ツリーであって、そのノードが（鍵、優先順位）ペアで関連しているものである。このツリーは、ノード鍵に関しては２進サーチ・ツリーであり（すなわち、どのノードに対しても、左（または右）のサブツリー内の鍵は、その鍵よりも小さい（または大きい）。）、そしてノード優先順位に関しては山（heap）である（すなわち、どのノードに関しても、その優先順位は、その子孫の優先順位よりも高い）。（鍵、優先順位）ペアのどの有限の集合も、トリープとしての固有の表現を有している。ランダム・トリープにおいては、優先順位は、十分大きな整然とした集合からランダムに引き出す（したがって、これらは、区別（distinct）できると仮定する）。

【００５４】シーデルとアラゴン〔2〕（Seidel and Aragon〔2〕）は、メンバーシップの照会、挿入、削除を、トリープ内に記憶された集合Ｓサイズの予測時間複素（complexity）対数をもって、処理するための単純なアルゴリズムを提示する。ランダム・トリープは、２－３ツリーと同様に、２－３ツリーに容易に変換することができる。これら手法の通信コストは、同様であるが、理由は、ランダム・トリープの予想される深さがその２－３ツリーの対応部分と類似しているからである。

【００５５】・　ランダム・トリープの主な利点は、それらの実現が、２－３ツリーの実現と比べはるかに簡単であることである。

・　ランダム・トリープの使用における短所は、これらの性能は、最悪の場合に保証されないことであり、例えば、あるユーザは、（確率は低いが）長い認証パスを得ることがある。

・　別の短所は、より強力な仮定がディレクトリに関して必要となることである。ランダム・トリープの分析は、敵（adversary）がトリープの正確な表現を知ることはない、ということに基づいている。証明書の状態を変更する能力をもつ信用できないディレクトリは、このシステムの計算作業および通信コストを増大させるおそれがある。

【００５６】本発明のシステムの動作は、図３に示した本発明の１実施形態を参照する１つの非限定の動作シーケンスにおいて例示する。

【００５７】一般的に言えば、ＣＡ、ディレクトリ、ユーザの手法において、次のステップを含む方法を提供する。

(a) ユーザは、ディレクトリに対し、少なくとも１つのアイテムのリストであって、ある集合内のその少なくとも１つのメンバーシップまたは非メンバーシップを認証するためのリストを提供し、(b) ディレクトリは、ユーザに対し、上記少なくとも１つのアイテムにより誘起された認証パス（単数または複数）を計算し送信し、そして(c) ユーザは、上記アイテムを検証する。

【００５８】さらにまた、ＣＡ、ディレクトリ、ユーザの手法において、次のステップから成る方法を提供する。上記ＣＡは、（ｉ）上記サーチ・ツリーを更新して、更新したノードを得るようにし、(ii) 前記更新されたノードにより誘起された認証パスを計算し、そして(iii) 少なくとも上記のルート修正されたノードをデジタル署名により認証し、(iv) 修正したパラメータを上記ディレクトリに送信すること、を実行し、上記ディレクトリは、（ｉ）上記の修正パラメータを適用することにより、認証されたディレクトリ・ルート値を得るようにし、(ii) 上記認証されたＣＡルート値が上記認証されたディレクトリ値と一致したか検証すること、を実行する。次に、この一般的な態様の具体的な説明をする。

【００５９】ＣＡ処理：

・証明書の作成：　ＣＡは、証明書データ（例えば、ユーザ名と公開鍵）、証明書連続番号、および満了日を含むメッセージを認証することにより証明書を生成する。

・初期化：　ＣＡは、最初は取り消されている証明書の集合に対して、上記のように２－３ツリーを作成する。ＣＡは、ツリー・ノード全ての値を計算し記憶し、そしてディレクトリに対し、取り消された証明書の連続番号の（記憶された）リストを、ツリー・ルート値、ツリーの高さおよびタイムスタンプを含む署名したメッセージと共に送る。

・更新：　ＣＡは、ツリーに対し証明書の挿入／削除を行うことによりツリーを更新する。各挿入／削除の後、全ての誘起されたノードを更新し、そしてそれにしたがって認証されたパスを計算する。ディレクトリを更新するには、ＣＡは、修正パラメータを送る。これは、例えば誘起されたノードのリスト、取引のリストである。実際には、修正パラメータは、ディレクトリがツリーをディレクトリ端において更新できるようにする情報であればどのような種類のものをも包含する。もちろん、ルートを認証することには、もちろん新たなルート値を含むだけでなく、他の認証された情報、例えばツリーの高さおよびタイムスタンプも同様に含むことができる。

【００６０】ディレクトリ処理：

・初期化：　最初は取り消されている証明書リストを受け取ったときに、ディレクトリは、ディレクトリ自身で２－３ツリー全体を計算し、そのルート値、ツリー高さ、およびタイムスタンプをチェックし、そしてこれらの値におけるＣＡの署名を検証する。

【００６１】・ＣＡの更新に対する応答：　ディレクトリは、ＣＡから受けた修正されたパラメータにしたがってツリーを更新する。これにより、再計算したパスと認証されたディレクトリ・ルートが生じる。これを行うことにより、ＣＡから受けた受信した認証されたルート値に対してその得たルート値をチェックし検証することにより、一致するか判定し、そして一致する場合には、この手順は、首尾良く終了する。この特定の実施形態では、上記のルート値、ツリー高さおよびタイムスタンプ全てが一致しなければならない（時間は、もちろん妥当なインターバル内である）。

【００６２】・ユーザの照会に対する応答：　ユーザの照会に回答するため、ディレクトリは、ユーザに対し、その認証したルート値、ツリー高さおよびタイムスタンプを供給する。

１．　その照会された証明書が取り消されている場合、そのルートから照会された証明書に対応するリーフまでのパスにおける各ノードに関して、ディレクトリは、ユーザにその値およびその子供の値を供給する。

２．　照会された証明書が取り消されていない（リストにない）場合、ディレクトリは、ユーザに対し、隣接する２つのリーフl_1，l_2へのパスを供給することにより、l_1（またはl_2）の値が照会された連続番号よりも小さくなる（または大きくなる）ようにする。

【００６３】ここで、通信コストを削減するため、ディレクトリは、ルートからのパス上のノード値を送る必要はないが、ユーザがサーチ・パス全体を計算するのに必要なもののみを送る必要がある。これは、図１を参照して例示したものであり、ここで想起されるように、Y5, H(6,6,Y)，H(7,8,Y)およびH(1,4,Y)のみが、Y5のサーチ・パスを認証するのに必要だったものである。

【００６４】ユーザ処理：ユーザは、まず初めに証明書に対するＣＡの署名を検証し、そして証明書の満了日をチェックする。次に、ユーザは、その証明書連続番号sをディレクトリに送ることにより照会を発する。照会に対するディレクトリからの回答を受け取ると、ユーザは、そのルート値、ツリー高さおよびタイムスタンプに対するＣＡの署名を検証する。

１．　ディレクトリが、照会に係る証明書が取り消されていると主張した場合、ユーザは、ハッシュ関数hを適用することにより、ディレクトリが供給したリーフからルートまでのパスをチェックする。

２．　ディレクトリが、その照会に係る証明書が取り消されていないと主張した場合、ユーザは、ディレクトリが供給した２つのパスをチェックし、そしてこれらが２－３ツリー内の値l_1，l_2をもつ２つの隣接リーフに至ることをチェックする。ユーザは、$l_1 <; s <; l_2$であることをチェックする。図２のＢに示すように、xを認証する（すなわち、xが取り消しリストに属していないことを主張する）ため、隣接メンバx1，x2を認証するサーチ・パスを送信する。ただし、x1 <; x <; x2である。

【００６５】上記手法においては、証明書が取り消されていなかったことを検証する通信コストは、証明書がリストにあることを検証するときの通信コストの２倍となることがある。これを克服するには、ツリーは、どのノードも連続した２つの連続番号に対応するように構築して、いずれの場合においても１つのパスのみを送るだけで済むようにできる。１つのツリー・ノードの値を保持するのに必要なビット数、すなわち、ハッシュ関数のセキュリティ・パラメータ（以下ではl_{hash}と記す）は、証明書の連続番号を保持するのに必要なビットの２倍以上であるため、これは、ツリー・サイズに影響を及ぼさない。これと関係して、想起されるように、“証明書”または“アイテム”は、とりわけ値のレンジを含む。

【００６６】次に、図４を参照すると、これは、本発明の別の実施形態によるシステム構成を示している。これでは、ある種のプロトコルが、短期証明書（short-term certificate）を使用することにより、取り消しシステムの必要性を回避している（例えば、証明書の所有者がある限られた損害を起こし得る場合には、マイクロペイメント・プロトコル（micropayments protocols）[13]）。これら証明書は、毎日発行し、そしてその発行日の終わりに満了する。実際には、さらに短い期間が望ましいが、主要な制限は、短期証明書が生じさせる証明機関の計算（ユーザ全てに対する証明書を毎日計算しなければならない）と通信（証明書はその所有者に送らなければならない）の増大に起因する。

【００６７】（ＣＲＳのような）オンライン／オフラインのデジタル署名手法は、ＣＡが実行しなければならない計算を低減させるけれども、通信コストを大きく減少させないが、それは、ＣＡが“異なった”メッセージを“異なった”ユーザに送らなければならず、ＣＡを通信のボトルネックとするからである。これは解決を必要とし、この解決では、ＣＡは、（例えば、新たなユーザと証明書が更新されていないユーザのみに関して）単純な計算を実行し、そして共通の更新メッセージを“全て”のユーザに送る。このメッセージを使えば、非取り消しの証明書をもつ正確に全てのユーザは、彼らの証明書の有効性を証明することができることになる。この実施形態に合うようにするには、証明書取り消し手法に対する簡単な修正を提案し、これにより、ＣＡが同一の更新メッセージを全てのユーザに送る効率的な証明書更新手法がもたらされる。この解決法においては、全ての証明書についての情報を備えたディレクトリ（図４参照）の存在を全く仮定していないが、ディレクトリが送出した最新のメッセージを保有することができるローカルのディレクトリの存在を仮定している。

【００６８】前と同じように、この手法は、上記に提示した証明機関が作成した取り消した証明書（あるいはさもなければ有効な証明書）のツリーに基づいている。ディレクトリから証明書を抽出する方法が全くないため、どのユーザも、初期の証明書を得て、そしてこれは、ＣＡのメッセージを使用して更新することができる。詳細には、ＣＡは、どの発行された証明書をも、その有効性を証明するパスで拡大し、そしてこれのみが、証明書の周期的に更新される部分である。

【００６９】全ての証明書を同時に更新するには、ＣＡは、ツリーのそのコピーを更新し、そして前の更新から変更したツリー・パス（誘起されたサブツリーの１つの非限定の形態を構成する）を発表する。（図５Ａを参照されたい）。非取り消しの証明書を保持する各ユーザは、好ましくはその自己パスと一致するパス上の最下位のノードvを位置づけることにより誘起されたツリーとその自己パスを交差させ、そしてvから（ルートまで）新たなノード値をコピーすることにより彼のパスを更新する。取り消された証明書を保持する全てのユーザは、関数hlの一方向性特性を壊さない限り、彼らのパスを更新することができない。図５Ｂに示したように、ユーザは、自己パスを誘起されたサブツリーと一致するようにし、そして最下位の不一致のノード（図５Ａにおいてドット１００で示す）を求める。行うべき残っていることは、上記の検出したノードからルートまでのノードを更新し、そしてそのルートを認証してこれをＣＡから送信された認証されたルート値に対し検証することである。この手順は、各ユーザに課される計算オーバーヘッドの点からは、明らかに非常にコスト的に効果がある。

【００７０】ＣＡ通信を低減するため、例えば証明書を１時間毎に更新する、というようにこの更新手法を使用することができる。これは、あるユーザが彼らの証明書の更新に遅れを生じることがあり、したがってローカル・ディレクトリは、（例えば在来のプロキシ・サーバを使用することにより）いくつかの最新の更新メッセージを蓄えておくべきであり、そしていくつかの集団が、更新をして（１日の更新メッセージを結合する）数日遅れたユーザが証明書の更新をすることを可能にする。

【００７１】この後者の詳細な説明は、より一般的には以下の通り定め、ＣＡ、ユーザ手法における請求項６記載の方法は、以下から成る。上記ＣＡは、(i)上記サーチ・ツリーを更新して、更新したノードを取得し、(ii)上記更新したノードが誘起した認証パスを計算し、(iii)少なくとも上記のルート修正したノードをデジタル署名により認証し、(iv)誘起したサブツリーを上記ユーザに送信すること、を実行し、上記ユーザは、(i)前記誘起されたサブツリーをユーザの自己パスと交差させ、そしてユーザ認証されたルート値を取得し、(ii)認証されたＣＡルート値が認証されたユーザ値と一致したことを検証すること、を実行する。

【００７２】当業者には容易に理解されるように、図３および図４の実施形態の実現は、どのような特定のハードウェアあるいはソフトウェア・アーキテクチャにも拘束されるものではない。したがって、非限定の例では、

ＣＡ、ディレクトリおよびユーザは、任意の利用可能な通信ネットワーク、例えばインターネットにより相互にリンクすることができる。別の非限定の例では、詳細な構成要素の各々は、特定の用途に依存して、必要でかつ適当な限り、例えば在来のＰＣコンピュータ、メインフレーム・コンピュータ、あるいはコンピュータのネットワークで実現できる。

【００７３】評価

以下においては、ＣＲＬと、ＣＲＳと、本発明のシステム／方法の１つの非限定の実施形態との通信コストを比較する。この分析に基づき、提案したシステムは、パラメータの変化に対しより強じんであり、そして他のものと比べより高い更新レートを可能にすることが示された。提案した手法の他の利点は、以下の通りである。

・　ＣＡは、ＣＲＳにおけるよりもより少ない機密しか保持する必要がない。

・　ＣＡ－ディレクトリ間通信が低いため、ＣＡは、ＣＡのコンピュータへの侵入に対し安全な低速通信回線を使ってディレクトリと通信するようにすることができる（このシステム・セキュリティは、ＣＡの機密を保護する能力に基づいたものである。）。

・　２－３ツリーの場合においては、更新するのにツリー全体を再計算する必要が決してない。これは、ＣＲＴと比べより高い更新レートを可能にする。

・　低いＣＡ－ディレクトリ間通信の別の結果として、ＣＡは、多くのディレクトリを更新することができ、通信ネットワークにおけるボトルネックを回避できる。

【００７４】通信コスト

我々が考察するパラメータは、以下の通りである。

・ｎ－証明書の見積もり総数（$n = 3,000,000$）

・ｋ－１つのＣＡが取り扱う証明書の見積もり平均数（$k = 30,000$）

・ｐ－満了前に取り消される証明書の見積もり割合（$p = 0.1$）。（証明書は１年間に発行され、したがって毎日取り消される証明書の数は、$n.p.l_{sn}/365$である、と仮定する。）

・ｑ－１日当たりに証明書の状態が照会される見積もり数（$q = 3,000,000$）

・Ｔ－１日当たりの更新数（$T = 1$）

・l_{sn}－証明書の連続番号を保持するのに必要なビット数（$l_{sn} = 20$）

・l_{stat}－証明書取り消し状態数Y_{365-i}およびN_0を保持するのに必要なビット数（$l_{stat} = 100$）

・l_{sig}－署名の長さ（$l_{sig} = 1,000$）

・l_{hash}－ハッシュ関数のセキュリティ・パラメータ（$l_{hash} = 128$）

n, k, p, q, T, l_{sn}, l_{stat}の値は、Micali〔18〕から取得し、そしてl_{sig}，l_{hash}は、本発明の手法に特有である。

【００７５】ＣＲＬコスト

・　ＣＲＬの毎日の更新コストは、$T.n.p = l_{sn}$であり、これは、各ＣＡが各更新においてＣＲＬ全体をディレクトリに送るためである。代替の更新手順では、ＣＡは、ディレクトリに対し、差リストのみ（前のＣＲＬに対し追加／除去する連続番号）を送り、これのコストは、$n.p.l_{sn}/365$である。

・　ＣＲＬの毎日の照会コストは、$q.p.k.l_{sn}$であるが、それは、各照会に関して、ディレクトリがＣＲＬ全体を照会側のユーザに送るからである。

【００７６】ＣＲＳコスト

・　ＣＲＳの毎日の更新コストは、$T.n.(l_{sn} + l_{stat})$であるが、それは、どの証明書に関しても、ＣＡは、l_{stat}ビットの証明書取り消し状態を送るからである。

・　ＣＲＳの毎日の照会コストは、$l_{stat}.q$である。

【００７７】提案した手法

ディレクトリを更新するためには、ＣＡは、$n.p.l_{sn}/365+T.l_{sig}$の毎日の総合の長さの差リストを送る。

・　ユーザの照会に回答するため、ディレクトリは、$2.\log_2(p \cdot k)$までの数を送り、各々がl_{hash}ビット長であり、全部で$2.q.l_{hash}.\log_2(p \cdot k)$ビットである。

【００７８】以下の表は、３つの手法による毎日の通信コスト（ビット）の見積もりを示している。

【００７９】

【表１】

	CRLコスト	CRSコスト	提案手法
毎日の更新（ＣＡ－ディレクトリ）	$6\text{-}10^8$	$3.6\text{-}10^8$	$1.7\text{-}10^4$
毎日の照会（ディレクトリ－ユーザ）	$1.8\text{-}10^{11}$	$3\text{-}10^8$	$7\text{-}10^5$

【００８０】この表に示したように、提案した手法のコストは、ＣＲＬのコストと比べ、ＣＡ－ディレクトリ間並びにディレクトリ－ユーザ間の両方の通信において低い。ＣＡ－ディレクトリ間コストは、対応するＣＲＳのコストよりもはるかに低いが、ディレクトリ－ユーザ間（したがって全て）の通信コストは増大している。尚、実際には、通信オーバーヘッドのため、ディレクトリ－ユーザ間通信におけるＣＲＳと本提案方法との間の差は、それ程でもない。

【００８１】本提案手法は、ＣＲＬおよびＣＲＳよりもパラメータの変化に対してより強じんである。何故なら、それらＣＲＬとＣＲＳは時間変化に拘束されるため、あるいは異なった実施の特定の必要によるため、そのような変化に対し強じんなシステムとすることが重要である。

【００８２】変化は、主に証明書の総数（ｎ）および更

新レート（T）に起きる。提案方法においては、nの変化は、pのファクタで加減している。Tの変化は、更新通信コストがnTではなくTに比例しているということにより適度にされている。図8は、3つの方法のCA－ディレクトリ間の更新通信コストがどのように更新レート（その他の全てのパラメータは一定とする）に依存しているかを示している。この更新通信コストは、CRSを1日に1回の更新に制限する（この更新レートを制限する別のファクタは、証明書が取り消されていなかったことを検証するためユーザが必要とする計算の量である。）。提案手法は、はるかに強じんであり、1時間に1回の更新でも許容できる。

【0083】以上、本発明についてある程度具体的に説明したが、特許請求の範囲に定めた本発明の範囲および要旨から逸脱せずに種々の変更および置換が行えることは理解されるべきである。

【図面の簡単な説明】

【図1】従来技術による認証ツリーを示す。

【図2】AとBは、本発明の1実施形態によるサーチ認証形ツリーを示す。

【図3】本発明の1実施形態によるシステム構成を示す。

【図4】本発明の別の実施形態によるシステム構成を示す。

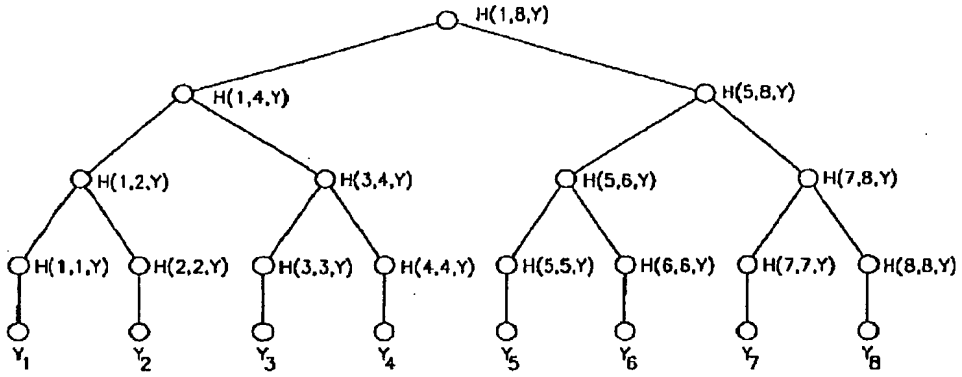
【図5】AとBは、サーチ認証形ツリーを図4の実施形態により更新する方法を示す。

【符号の説明】

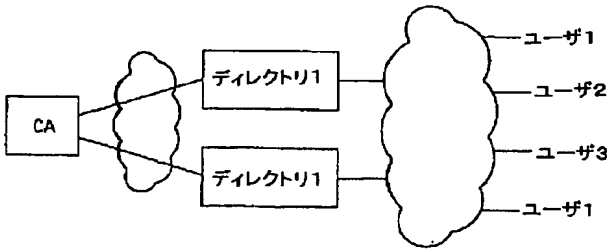
Y_1－Y_8　クレジットカード

c1－c7　証明書の連続番号

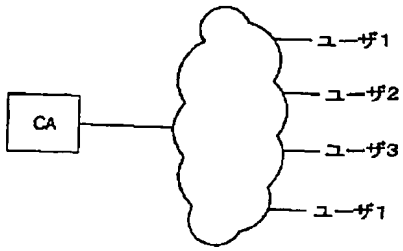
【図1】



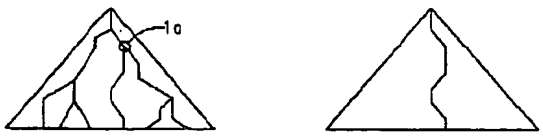
【図3】



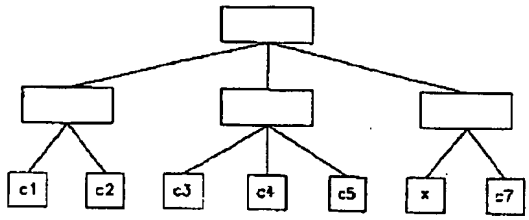
【図4】



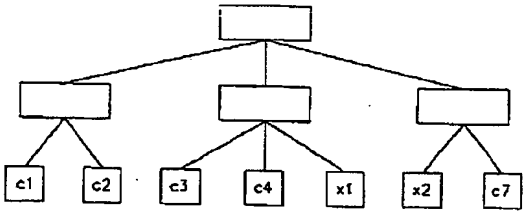
【図5】



【図2】



A



B

【外国語明細書】

FIELD OF THE INVENTION

The present invention is in the general field of digital signature for authentication purposes.

BACKGROUND OF THE INVENTION

The wide use of public key cryptography requires the ability to verify the authenticity of public keys. This is achieved through the use of certificates (that serve as a mean for transferring trust) in a *Public Key Infrastructure* (PKI). A certificate is a message signed by a publicly trusted authority (the certification authority, whose public key authenticity may be provided by other means) which includes a public key and additional data, such as expiration date, serial number and information regarding the key and the subject entity.

When a certificate is issued, its validity is limited by an expiration date. However, there are circumstances (such as when a private key is revealed, or when a key holder changes affiliation or position) where a certificate must be revoked prior to its expiration date. Thus, the existence of a certificate is a necessary but not sufficient evidence for its validity, and a mechanism for determining whether a certificate was revoked is needed.

A typical application is a credit card system where the credit company may revoke a credit card, temporarily or permanently, prior to its expiration, e.g. when a card is reported stolen or according to its user's bank account balance.

PRIOR ART DISCUSSION:

CertificateRevocation List (CRL)

A CRL is a signed list issued by the CA identifying all revoked certificates by their serial numbers. The list is concatenated with a time stamp (as an indication of its freshness) and signed by the CA that originally issued the certificates. The CRLs are sent to the directory on a periodic basis, even if there are no changes, to prevent the malicious replay of old CRLs instead of new CRLs.

As an answer to a query, the directory supplies the most updated CRL (the complete CRL is sent to the merchant).

- The main advantage of the scheme is its simplicity.
- The main disadvantage of the scheme is its high directory-to-user communication costs (since CRLs may get very long). Another disadvantage is that a user may not hold a succinct proof for the validity of his certificate.

A reasonable validity expiration period should be chosen for certificates. If the expiration period is short, resources are wasted reissuing certificates. If the expiration period is long, the CRL may get long, causing high communication costs and difficulties in CRL management. Kaufman *et al.* [15, Section 7.7.3] suggested reissuing all certificates whenever the CRL grows beyond some limit. In their proposal, certificates are marked by a serial number instead of an expiration date. (Serial numbers are incremented for each issued certificate. Serial numbers are not reused even when all certificates are reissued.) The CRL contains a field indicating the *first valid* certificate. When all certificates are reissued, the CRL first valid certificate field is updated to contain the serial number of the first reissued certificate.

Certificate Revocation System

Micali [18] suggested the Certificate Revocation system (CRS) in order to improve the CRL communication costs. The underlying idea is to sign a message for every certificate stating whether it was revoked or not, and to use an off-line/on-line signature scheme [11] to reduce the cost of periodically updating these signatures.

To create a certificate, the CA associates with each certificate two numbers (Y_{365} and N) that are signed along with the 'traditional' certificate data. For each certificate, the CA chooses (pseudo) randomly two numbers N_0 Y_0 and computes (using a one-way function f) $Y_{365} = f^{365}(Y_0)$ and $N = f(N_o)$. (Actually, a stronger assumption on f is required, e.g. that f is one-way on its iterates, i.e. that given $y = f^{i}(x)$ it is infeasible to find x' such that $y = f(x')$. This is automatically guaranteed if f is a one-way permutation.)

The directory is updated daily by the CA sending it a number C for each certificate as follows:

1. For a non-revoked certificate, the CA reveals one application of f, i.e. $C = Y_{365-i} = f^{365-i}(Y_0)$, where i is a daily incremented counter, $i = 0$ on the date of issue.

2. For a revoked certificate, $C = N_0$

Thus the most updated value for C serves as a short proof (that certificate x was or was not revoked) that the directory may present in reply to a user query x.

- The advantage of CRS over CRL is in its query communication costs. Based on Federal PKI (Public Key Infrastructure) estimates, Micali [18] showed that although the daily update of the CRS is more expensive than a CRL update, the cost of CRS querying is much lower. He

estimated the resulting in 900 fold improvement in total communication costs over CRLs.

Another advantage of CRS is that each user may hold a succinct transferable proof of the validity of his certificate. Directory accesses are saved when users hold such proofs and presents them along with their certificates.

- The main disadvantage of this system is the increase in the CA-to-directory communication (it is of the same magnitude as directory-to-users communication, where the existence of a directory is supposed to decrease the CA's communication). Moreover, since the CA's communication costs are proportional to the directory update rate, CA-to-directory communication costs limit the directory update rate.

The complexity of verifying that a certificate was not revoked is also proportional to the update rate. For example, for an update once an hour, a user may have to apply the function, f, 365 x 24 = 8760 times in order to verify that a certificate was not revoked, making it the dominant factor in verification.

Certificate Revocation Trees

Kocher [16] suggested the use of Certificate Revocation Trees (CRT) referred to also as authentication tree, in order to enable the verifier of a certificate to get a short proof that the certificate was not revoked. A CRT is a hash tree with leaves corresponding to a set of statements about certificate serial number X issued by a CA, CA_X. The set of statements is produced from the set of revoked certificates of every CA. It provides the information whether a certificate X is revoked or not (or whether its status is unknown to the CRT issuer). There are two types of statements: specifying ranges of unknown CAs, and, specifying certificates range of which only the lower certificate is

revoked. For instance, if CA_I revoked two certificates, $X1 < X2$, than one of the statements is:

$$if\ CA_x = CA_I \text{ and } X_I \leq X < X_2 \ \ then\ X\ is\ revoked\ if\ X = v$$

To produce the CRT, the CRT issuer builds a binary hash tree [17] with leaves corresponding to the above statements

A proof for a certificate status is a path in the hash tree, from the root to the appropriate leaf (statement) specifying for each node on the path the values of its children.

- The main advantages of CRT over CRL are that the entire CRL is not needed for verifying a specific certificate and that a user may hold a succinct proof of the validity of his certificate.

- The main disadvantage of CRT is in the computational work needed to update the CRT. Any change in the set of revoked certificates may result in re-computation of the *entire* CRT.

LIST OF CITED PRIOR ART

U.S. patent 4,309,569 (hereinafter the *Merkle* patent)

[1] A. V. Aho, J. E. Hopcroft, J. D. Ullman. "Data Structures and Algorithms". Addison-Wesley, 1983.

[2] R.G. Seidel., C.R. Aragon "Randomized Search Trees". Proc. 30th Annual IEEE Symp. on Foundations of Computer Science, pp. 540-545, 1989.

[6] M. Bellare, P.Rogaway. "Collision-Resistant Hashing: Towards Making UOWHFs Practical". Advances in Cryptology – CRYPTO '97, Lecture Notes in Computer Science, Springer-Verlag, 1997.

[11] S. Even, O. Goldreich, S. Micali. "On-Line/Off-Line Digital Signatures". Journal of Cryptology, Springer-Verlag, Vol. 9 pp. 35-67, 1996.

[12] O. Goldreich, S. Goldwasser, and S. Halevi,. "Collision-Free Hashing from Lattice Problems". *ECCC*, TR96-042, 1996.
http: //www. eccc. unitrier. de/eccc/

[13] A. Herzberg, H. Yochai. "Mini-Pay: Charging per Click on the Web". Proc. 6th International World Wide Web Conference, 1997.
http: //www6. nttlabs. com/

[15] C. Kaufman, R. Perlman, M. Speciner. "Network Security. Private Communiation in a Public World". Prentice Hall series in networking and distributing systems, 1995.

[16] P. Koeher. "A Quick Introduction to Certificate Revocation Trees (CRTs)".
http: //www. valicert. com/company/crt. html

[17] R. C. Merkle. "A Certified Digital Signature". Proc. Crypto '89, Lecture Notes in Computer Science 435, pp. 234-246, Springer-Verlag, 1989.

[18] S. Micali. "Efficient Certificate revocation". Technical Memo MIT/LCS/TM542b, 1996.

GLOSSARY

There follows glossary of terms some of which conventional and others have been coined:

Certification Authority (CA) - A trusted party, already having a certified public key, responsible for establishing and vouching for the authenticity of public keys and/or other information such as credit card numbers.

A CA preferably, but not necessarily, does not provide on-line certificate information services to users. Instead, it updates a directory on a periodic basis). As will be shown below in some embodiments directories are not used.

A CA issues certificates for users by a message containing the certificate serial number relevant data and an expiration date. The certificate is sent to a directory and/or given to the user. The CA may revoke a certificate prior to its expiration date. *Certificate* is by no means bound to the latter definition and may encompass data pertain to e.g. one or more (such as range of) public key(s), credit card number(s), and others; presented either in explicit form or after having been subject to a function such as encoding or encryption. (the term *item* and *certificate* are used in the specification interchangeably)

Directory - : One or more non-trusted parties that get updated certificate revocation information from the CA and serve as a certificate database accessible by the users.

User - A non-trusted party that receives its certificate from the CA and issues queries for certificate information. User should be construed as encompassing among others:

(i) a merchant who queries the validity of other users' certificates,

(ii) a user who gets proof of the validity of his/her certificate for using it *vis-a-vis* other users.

Search tree – A well known data structure that is associated with search scheme which enables to construct a search path in the tree, from the root to a sought item (associated with a leaf). The search path exploits search values that reside in the tree nodes and possibly also in the links. Search tree is inherently designed to handle update transactions (i.e. delete and/or insert items to the tree). Typical, yet not exclusive, examples of search trees being: 2-3 tree, Btree, Btree+, TriS, treaps and others.

Update Transactions - Insert new item to a tree; delete existing item in a tree.

Authentication Tree – a rooted tree where each internal node authenticates the values of its children by means of a cryptographic hash function and the root is authenticated by means of a digital signature. Typical, yet not exclusive, example is illustrated in the *Merkle* patent.

Cryptographic hash function- includes:

(i) *collision intractable function* h() such that it is computationally essentially infeasible to find $y \neq x$ satisfying $h(x) = h(y)$. Typical, yet not exclusive example is illustrated in the *Merkle* patent; or

(ii) *universal one way hash function* h() such that there exists a family of functions h() such that for every x and random h() from the family, it is computationally essentially infeasible to find $y \neq x$ satisfying $h(x) = h(y)$. (for detailed discussion in (I) and (II), see [6]).

SUMMARY OF THE INVENTION

There is, accordingly, a need in the art for eliminating or substantially reducing the drawbacks associated with hitherto known techniques by providing a novel technique for authenticating items.

The present invention incorporates the utilization of conventional authentication trees as well as conventional search trees such as 2-3 tree or Btree. The utilization of search trees enables to authenticate an item (or items) whilst obviating the need to transmit a large amount of data to this end. The utilization of the authentication tree, according to the prior art, enables to transmit a series of revoked, (or otherwise) valid items.

The major drawback of using an authentication tree, e.g. of the kind disclosed in the *Merkle* patent, arises when the latter is subject to modification transactions. The latter bring about new arrangement of items in the leaves and, consequently, (as will be exemplified below), necessitates the modification of the values of multitude nodes (hereinafter modified nodes) in the tree.

Not only is an extensive computation required in order to update the values of the modified nodes, but also by utilizing an authentication high communication overhead is imposed when the multitude values of said modified nodes are transmitted over the communication network, e.g. from the CA to the directory. Considering that such modification may occur quiet frequently, the specified overhead renders the use of prior art authentication trees commercially infeasible.

According to the invention, a conventional authentication tree is "superimposed" on conventional search tree (bringing about authentication search tree) benefiting thus both from the inherent advantages of the authentication tree insofar as authenticating items is concerned and from the limited changes that are imposed on the tree nodes due to the search tree structure.

Accordingly, the present invention provides for a memory containing an authenticated search tree that serves for authenticating membership or non membership of items in a set; the authenticated search tree, comprising:

a search tree having nodes and leaves and having associated therewith a search scheme; the nodes including dynamic search values and the leaves including items of said set; the nodes are associated, each, with a cryptographic hash function value that is produced by applying a cryptographic hash function to at least: (I) the cryptographic hash values of the children nodes and (II) the dynamic search value of said node;

at least the root node of said authenticated search tree is authenticated by a digital signature.

Still further the invention provides for a method for authenticating membership or non membership of items in a set; comprising:

(i) providing an authenticated search tree of the kind specified;

(ii) authenticating at least one item of said set by computing the authentication path as induced by said at least one item and the root.

Still further the invention provides for a method for updating at least one item of a set in an authenticated search tree, comprising:

(i) providing a search authenticated tree of the kind specified;;

(ii) updating said search tree so as to obtain updated nodes;

(iii) computing an authentication path as induced by said updated nodes; and

(iv) authenticating at least said root modified node by a digital signature.

It should be noted that the specified order does not necessarily imply that in iterative procedure all the steps are performed in each iteration. Thus for example the steps (ii) and (iii) may be performed in each iteration and step (iv) may be applied once at the last iteration. This, likewise, applies to the other aspects of method and system as described herein.

The invention further provides a system for authenticating/updating *mutatis mutandis*.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention will now be described, by way of example only with reference to the accompanying drawings, in which:

Fig. 1 illustrates an authentication tree according to the prior art;

Figs. 2A-B illustrate a search authenticated tree according to one embodiment of the invention;

Fig. 3 illustrates a system configuration according to one embodiment of the invention;

Fig. 4 illustrates a system configuration according to another embodiment of the invention; and

Fig. 5A-B illustrate a manner in which a search authenticated tree is updated according to the embodiment of Fig. 4.

DESCRIPTION OF SPECIFIC EMBODIMENTS

Attention is first directed to Fig. 1 illustrating an authentication tree according to the prior art e.g as disclosed in the specified *Merkle* patent, the contents of which are incorporated herein by reference.

Consider, for example, that certificates Y_1 to Y_8 stand for a certificate list (CL) of all valid credit cards. Now, a user wants to use his credit card Y_5 in a commercial transaction *vis-a-vis* a merchant. The merchant addresses a directory that holds the authentication tree (i.e. authentication tree in respect of valid credit cards Y_1 to Y_8) of the kind disclosed in Fig. 1. It is recalled that the directory is an un-trusted party and therefore the merchant wants to verify that Y_5 indeed appears in the tree.

The collision intractable function h() serves for authenticating item(s) (credit cards) Y_1 to Y_8 sorted according to credit card number, e.g. in ascending order. Thus, in order to authenticate Y_5, it is sufficient for the directory to transmit to the merchant tree leaf and node values Y_5, H(6,6,Y), H(7,8,Y) and H(1,4,Y), assuming that the root value H(1,8,Y) was previously authenticated, e.g. using a digital signature. Of course, additional tree values may be transmitted but as will be appreciated from the description below transmitting additional tree values is absolutely redundant.

Thus, in order to authenticate Y_5, the merchant (knowing *a priori* H()) calculates the authentication path, namely, H(5,5,Y) (on the basis of Y_5) and on the basis of H(5,5,Y) and the so received H(6,6,Y), the merchant calculates H(5,6,Y). The latter, along with so received H(7,8,Y) give rise to H(5,8,Y). The latter along with the so received H(1,4,Y) give rise to H(1,8,Y) which is subject to PKI technique (e.g. applying the public key *n*), and the result is compared to the previously authenticated H(1,8,Y) value and in the case of match, it is assured that the item Y_5 belongs to the list of valid credit cards.

The advantage of the authentication tree is, of course, that only few tree node values were transmitted to the user which could nevertheless authenticate the item Y_5 of interest. As will be explained in below, the specified description for authenticating items in respect of prior art authentication tree applies also to the search authenticated tree of the invention.

The major drawback of the authentication tree of Fig. 1 arises when the latter is subject to modify transaction, e.g. when new credit card is added to the list at the CA. Suppose that new item $Y_{4'}$ such that $Y_4 < Y_{4'} < Y_5$ is added. The resulting authentication tree (not shown) will necessitate extensive update of most of the nodes in the tree and undue transmission overhead of the updated information, which is obviously undesired, particularly when bearing in mind that the rate of updating the CA with new items is as a rule quiet high.

The advantages and disadvantages equally apply when considering a certificate revocation list (CRL) which holds the invalidated or revoced items (e.g. invalid credit cards

Considering now an exemplary search authenticated tree according to one embodiment of the invention utilizing e.g. a 2-3 search tree with a CRL (e.g. a list of revoked credit cards held at the leaves, See Fig. 2A-B.)

In this connection it should be noted that the invention is, by no means, bound to the actual realization of the search tree and any known technique that is utilized to this end is applicable, all as required and appropriate depending upon the particular application. Thus, by way of non limiting example, any manner of holding the items in the leaves is applicable, e.g. as records, link list, tree, of blocks (in the case of long item) etc. This statement is likewise valid to the authentication tree.

Thus, by this particular embodiment, a 2-3 tree is maintained with leaves corresponding to the revoked certificates' serial numbers (c1-c7) in increasing order. (In a 2-3 tree every interior node has two or three children and the paths from root to leaves have the same length). Testing membership and modifying, i.e. inserting, deleting or updating a single element are done in logarithmic time, where the modification affects only the nodes on the modification path. For a detailed presentation of 2-3 trees see [1, pp.169-180].) The property of 2-3 trees is that test and modification involve only changes to nodes on a search path, i.e. every change is local and the number of affected paths is small.

The tree may be created either by inserting the serial numbers of the revoked certificates one by one into an initially empty 2-3 tree, or, by sorting the list of serial numbers and building a degree 2 tree with leaves corresponding to the serial numbers in the sorted list (because the communication complexity is minimal when the tree is of degree 2).

Every tree node is assigned a value according to the following procedure:

- Each leaf stores a revoked certificate serial number as its value.
- The value of an internal node is computed by applying the cryptographic hash function H() to the values of its children and to at least the dynamic search values of the internal node (which encompasses also link, whenever applicable). Whilst it is not obligatory, the cryptographic one way hash function H() may also be applied to information, other than the dynamic search values that are associated with the node, e.g. information relevant for balancing the tree etc.

Unlike the collision intractable function, applying the universal one way hash function to the internal nodes in the manner specified, necessitates utilization of unique function for each node. For the latter case, it is required to authenticate in addition to the above referred to values of the children and the dynamic search values of the internal node, also the unique value of the function that is associated to the internal node.

There follows now a description that pertains to modifying the search authenticated tree according to one embodiment of the invention.

Thus, in order to delete an item, a conventional 2-3 delete item step is executed, namely:

1. Delete each expired certificate serial number from the 2-3 tree, updating the values of the nodes on the deletion path.

Likewise, in order to insert an item, a conventional 2-3 insert item step is executed, namely:

2. Insert each newly revoked certificate serial number into the tree, updating the values of the nodes on the insertion path.

During tree update, some new nodes may be created or some nodes may be deleted due to the balancing of the 2-3 tree. These nodes occur only on the search path for an inserted/deleted node (hereinafter: modified node).

The certification authority authenticates the tree by authenticating the root and to this end, only the search path that is induced by the modified nodes should be computed.

For a simpler implementation of thesearch authenticated tree, other trees, e.g. random treaps [2], may be used instead of 2-3 trees. Treaps are binary trees whose nodes are associated with (key, priority) pairs. The tree is a binary search tree with respect to node keys (i.e. for every node the keys in its left (resp. right) subtrees are small (resp. greater) than its key); and a heap with respect to node priorities (i.e. for every node its priority s higher than its descendents' priorities). Every finite set of (key, priority) pairs has a unique representation as a treap. In *random treaps*, priorities are drawn at random from a large enough ordered set (thus, they are assumed to be distinct).

Seidel and Aragon [2] present simple algorithms for membership queries, insert and delete operations with expected time complexity logarithmic in the size of the set S stored in the treap. Random treaps may be easily converted into authenticated search data structures similarly to 2-3 trees. The communication costs of these schmes is similar since the expected depth of a random treap is similar to its 2-3 tree counterpart.

- The main advantage of random treaps is that their implementation is much more simple than the implementation of 2-3 trees.
- A drawback of using random treaps is that their performance is not guaranteed in worst case, e.g. some users may (with low probability) get long authentication paths.
- Another drawback is that a stronger assumption is needed with respect to the directory. The analysis of random treaps is based on the fact that the adversary does not know the exact representation of a treap. A dishonest directory with ability to change the status of certificates may increase the computational work and communication costs of the system.

The operation of a system of the invention will be exemplified in one non-limiting sequence of operation which refers to an embodiment of the invention as depicted in Fig. 3.

Generally speaking there is provided a method in a CA, directory, user scheme, including the steps of:

(a) the user providing to a directory a list of at least one item for authenticating membership or non membership of said at least one item in a set;

(b) the directory computing and transmitting to a user the authentication path(s) as induced by said at least one item; the directory further transmitting said authenticated root; and

(c) the user verifying said items.

Still further there is provided a method in a CA directory user scheme comprising the steps of:

the CA executing:

(i) updating said search tree so as to obtain updated nodes;

(ii) computing an authentication path as induced by said updated nodes; and

(iii) authenticating at least said root modified node by a digital signature;

(iv) transmitting modified parameters to said directory;

the directory executing:

(i) applying said modification parameters, so as to obtain authenticated directory root value;

(ii) verifying that the authenticated CA root value matched the authenticated directory value.

A specific description of the general aspect above will now be described:

CA Operations

- **Creating certificates:** The CA produces a certificate by authenticating a message containing certificate data (e.g. user name and public key), certificate serial number and expiration date.

- **Initialization:** The CA creates the 2-3 tree, as above, for the set of initially revoked certificates. It computes and stores the values of all the tree nodes and sends to the directory the (sorted) list of revoked certificate serial numbers along with a signed message containing the tree root value, the tree height and a time stamp.

- **Updating:** The CA updates the tree by inserting/deleting certificates from it. After each insertion/deletion, all induced nodes are updated and the authenticated path is calculated accordingly. To update the directory, the CA sends a modification parameters. The latter may be for example the list of induced nodes, the list of the transactions. In fact modification parameter encompass any kind of information that enables the directory to update the tree at the directory end. Of course, authenticating the root encompasses of course the new root value but may likewise include other authenticated information e.g. tree height and time stamp.

Directory operations:

- **Initialization:** Upon receiving the initial revoked certificates list, the directory computes by itself the whole 2-3 tree, checks the root value, tree height and time stamp, and verifies the CA's signature on these values.

- **Response to CA's update:** The directory updates the tree according to the modified parameters received from the CA. This results in recomputed path and authenticated directory root. Having done so it checks verifies the so obtained root value vis-a-vis the received authenticated root value as received from the CA in order to determine match, in which case the procedure terminates successfully. By this particular embodiment the root value, tree height and time stamp, all have to match (the time, of course, within reasonable interval).

- **Response to user's queries:** To answer a user query, the directory supplies the user with the authenticated root value, tree height and time stamp.

1. If the queried certificate is revoked, for each node in the path from the root to the leaf corresponding to the queried certificate, the directory supplies the user its value and its children values.

2. If the queried certificate is not revoked (not in the list), the directory supplies the user the paths to two neighbouring leaves l_1, l_2 such that the value of l_1 (resp. l_2) is smaller (resp. larger) than the queried serial number.

Note that to reduce the communication costs, the directory need not send the node values on the path from root, but only those which are required for the user to compute the entire search path. The latter was exemplified in reference to Fig. 1 where, as recalled, only Y5, H(6,6,Y), H(7,8,Y) and H(1,4,Y) were required in order to authenticate the search path of Y5.

User Operations:

The user first verifies the CA's signature on the certificate and checks the certificate expiration date. Then, the user issues a query by sending the directory the certificate serial number s. Upon receiving the directory's answer to a query, the user verifies the CA's signature on the root value, tree height and time stamp.

1. If the directory claims the queried certificate is revoked, the user checks the leaf to root path supplied by the directory by applying the hash function h.

2. If the directory claims the queried certificate is not revoked, the user checks the two paths supplied by the directory and checks that they lead to two adjacent leaves in the 2-3 tree, with values l_1, l_2. The user checks

that $l_1 < s < l_2$. As shown in Fig. 2B for authenticating x (i.e. claiming that x does not belong to the revocation list) the search path that authenticate the adjacent members *x1* and *x2* are transmitted, where $x1 < x < x2$.

In the above scheme, the communication costs of verifying that a certificate was not revoked may be twice the communication costs of verifying that a certificate is in the list. To overcome this, the tree may be built such that every node corresponds to two consecutive serial number thus having to send only one path in either case. Since the number of bits needed for holding the value of a tree node, i.e. the hash function security parameter (l_{hash} in the notation below) is more than twice the bits needed for holding a certificate serial number, this does not influence the tree size. In this connection it is recalled that *certificate* or *item* embrace, amongst the other, range of values.

Attention is now drawn to Fig. 4 illustrating a system configuration according to another embodiment of the invention. Thus, some protocols avoid the need for a revocation system by using short-term certificates. (e.g. micropayments protocols when a certificate owner may cause a limited damage [13]). These certificates are issued daily and expire at the end of the day of issue. Actually, even shorter periods are desired and the main limit is due to the increase in the certification authority computation (certificates for all users have to be computed daily) and communication (certificates should be sent to their owners) short-term certificates cause.

An on-line/off-line digital signature scheme (like CRS) will reduce the computation the CA has to perform, but, it will not reduce significantly the communication costs, since the CA has to send *different* messages to *different* users, making the CA a communication bottleneck. This calls for a solution where the CA performs a simple computation (say, concerning only new users and users whose certificates are not renewed) and sends a common update message to *all* users. Using this message, exactly all users with non-revoked certificates should be able to prove the validity of their

certificates. To meet the latter embodiment, a simple modification to the certificate revocation scheme is proposed to yield an efficient certificate update scheme in which the CA sends the same update message to all users. In this solution there is no assumption of the existence of a directory (See Fig. 4) with information about all certificates, but of local directories that may hold the latest messages that are sent by the directory.

As before, the scheme is based on a tree of revoked certificates (or, otherwise, valid certificates) created by the certification authority presented above. Since there is no way to extract certificates from a directory, every user gets an initial certificate that may be updated using the CA's messages. Specifically, the CA augments every issued certificate with the path proving its validity, this is the only part of the certificate that is updated periodically.

To update all certificates simultaneously, the CA updates its copy of the tree, and publishes the tree paths that where changed since the previous update (constituting one, non limiting form of induced sub-tree). (see Fig. 5A). Every user holding a non-revoked certificate intersects its self path with the induced tree preferably by locating the lowest node, v, on a path that coincides with the self path, and updates his path by copying the new node values from v up to the root) . All users holding a revoked certificate can not update their path, unless they crack the one way characteristics of the function *hl*. As shown in Fig. 5B the user bring into coincidence the self path with the induced sub tree and seeks for the lower most discrepancy node (desgnated by dot 100 in Fig. 5A). What remains to be done is to update the nodes from the so detected node to the root and to authenticate the root and verify it vis-avis the authenticated root value transmitted from the CA. The latter procedure is obviously very cost effective in terms of the computation overhead that is posed on each user.

Since the CA communication is reduced, one may use this update scheme for, say, updating certificates once every hour. This may cause some users to lag in updating their certificates, and the local directories should save several latest update messages (e.g. using conventional proxy servers), and

some aggregate updates (combining update messages of a day) enabling users that lag several days to update their certificates.

The latter specific description is defined more generally as follows, a method according to Claim 6, in a CA user scheme comprising:

the CA executing:

(i) updating said search tree so as to obtain updated nodes;

(ii) computing an authentication path as induced by said updated nodes; and

(iii) authenticating at least said root modified node by a digital signature;

(iv) transmitting induced sub-tree to said user;

the user executing:

(i) intersecting said induced sub-tree with user self path and obtaining user authenticated root value;

(ii) verifying that the authenticated CA root value matched the authenticated user value.

Those versed in the art will readily appreciate that the realization of the embodiments of Figs. 3 and 4 are not bound to any specific hardware and/or software architecture. Thus, by way of non limiting example, the CA, directory and user may be interlinked by any available communication network, e.g. the Internet. By way of another non limiting example each of the specified constituents may be implemented on e.g. conventional P.C. computer, mainframe computer or network of computers, all as required and appropriate, depending upon the particular application.

Evaluation

In the following, the communication costs of CRL, CRS and one, non limiting, embodiment of a system/method of the invention are compared, Basing on this analysis, there is shown that the proposed system is more robust to changes in parameters, and allows higher update rates than the other.

Other advantages of the proposed scheme are.

- The CA has to keep a smaller secret than in CRS.
- Since CA-to-directory communication is low, the CA may communicate with the directory using a slow communication line secured against breaking into the CA's computer (the system security is based on the ability to protect the CA's secrets).
- In the case of a 2-3 tree, there is never a need to re-compute the entire tree to update it. This allows higher update rates than CRT.
- Another consequence of the low CA-to-directory communication is that a CA may update many directories, avoiding bottlenecks in the communication network.

Communication Costs

The parameters we consider are:

- n - Estimated total number of certificates ($n = 3,000,000$).
- k - Estimated average number of certificates handled by a CA ($k = 30,000$)
- p - Estimated fraction of certificates that will be revoked prior to their expiration ($p = 0.1$). (It is assumed that certificates are issued for one year, thus, the number of certificates revoked daily is $\frac{n.p.l_{sn}}{365}$.
- q - Estimated number of certificate status queries issued per day
- ($q = 3,000,000$).
- T – Number of updates per day ($T = 1$).
- l_{sn} - Number of bits needed to hold a certificate serial number ($l_{sn} = 20$).
- l_{stat} – Number of bits needed to hold the certificate revocation status numbers Y_{365-i} and N_0 ($l_{stat} = 100$).

- l_{sig} – Length of signature (l_{sig} = *1,000*).

- l_{hash} – Security parameter for the hash function (l_{hash} = *128*).

Values for n,k,p,q,T,l_{sn} l_{stat} are taken from Micali [18], l_{sig} and l_{hash} are specific to our scheme.

CRL Costs

- The CRL daily update cost is $T.n.p.=l_{sn}$ since each CA sends the whole CRL to the directory in each update. An alternative update procedure where the CA sends to the directory only a difference list (which serial numbers to add/remove from the previous CRL) costs:

$$\frac{n.p.l_{sn}}{365}.$$

- The CRL daily query costs is $q.p.k.l_{sn}$ since for every query the directory sends the whole CRL to the querying user.

CRL Cost

- The CRS daily update cost is $T.n.(l_{sn} + l_{stat})$ since for every certificate the CA sends l_{stat} bits of certificate revocation status.
- The CRS daily query cost is $l_{stat}.q$.

The proposed scheme

To update the directory, the CA sends the difference lists of total daily length of $\frac{n.p.kn}{365} + T.l_{sig}$

- To answer a user's query, the directory sends up to $2 \cdot \log_2 (p \cdot k)$ numbers, each l_{hash} bits long, totaling $2.q.l_{hash}.\log_2 (p \cdot k)$ bits.

The following table shows the estimated daily communications costs (in bits) according to the three schemes.

	CRL Costs	CRS Costs	Proposed Scheme
Daily update (CA-directory)	$6\text{-}10^6$	$3.6\text{-}10^8$	$1.7\text{-}10^4$
Daily queries (Directory-users)	$1.8\text{-}10^{11}$	$3\text{-}10^8$	$7\text{-}10^9$

As shown in the table, the proposed scheme costs are lower than CRL costs both in CA-to-directory and in directory-to-users communication. The CA-to-directory costs are much lower than the corresponding CRS costs but, the directory-to-user (and thus the over all) communication costs are increased. Note that in practice, due to communication overheads, the difference between CRS and the proposed method in Directory-to-users communication may be insignificant.

The proposed scheme is more robust to changes in parameters than CRL and CRS. Since these are bound to change in time or due to the specific needs of different implementations, it is important to have a system that is robust to such changes.

Changes will occur mainly in the total number of certificates (n) and the update rate (T). In the proposed method, changes in n are moderated by a factor of p. Changes in T are moderated by the fact that the update communication costs are not proportional to nT but to T. Figure 8 shows how the CA-to-directory update communication costs of the three methods depend on the update rate (all other parameters are held constant). The update communication costs limit CRS to about one update a day (Another factor that limits the update rate is the amount of computation needed by a user in order to verify that a certificate was not revoked). The proposed scheme is much more robust, even allowing once per hour updates.

The present invention has been described with a certain degree of particularity, but it should be understood that various modifications and alterations may be made without departing from the scope or spirit of the invention as defined by the following claims:

CLAIMS:

1. A memory containing an authenticated search tree that serves for authenticating membership or non membership of items in a set; the authenticated search tree, comprising:

a search tree having nodes and leaves and having associated therewith a search scheme; the nodes including dynamic search values and the leaves including items of said set; the nodes are associated, each, with a cryptographic hash function value that is produced by applying a cryptographic hash function to at least: (I) the cryptographic hash values of the children nodes and (II) the dynamic search value of said node;

at least the root node of said authenticated search tree is authenticated by a digital signature.

2. A search authenticated tree wherein said cryptographic hash function being of the universal one way function type, and wherein said cryptographic one way function is further applied to a universal one way function that is unique to each internal node.

3. A search authenticated tree of Claim 1, wherein said search tree being Btree.

4. A search authenticated tree of Claim 1, wherein said search tree being 2-3 tree.

5. A method for authenticating membership or non membership of items in a set; comprising:

(i) providing an authenticated search tree as defined in Claim 1;

(ii) authenticating at least one item of said set by computing the authentication path as induced by said at least one item and the root.

6. A method for updating at least one item of a set in an authenticated search tree, comprising:

(i) providing a search authenticated tree as defined in Claim 1;

(ii) updating said search tree so as to obtain updated nodes;

(iii) computing an authentication path as induced by said updated nodes; and

(iv) authenticating at least said root modified node by a digital signature.

7. A method according to Claim 5, in a CA, directory, user scheme, wherein said step (ii), includes:

(a) the user providing to a directory a list of at least one item for authenticating membership or non membership of said at least one item in a set;

(b) the directory computing and transmitting to a user the authentication path(s) as induced by said at least one item; the directory further transmitting said authenticated root; and

(c) the user verifying said items.

8. A method according to Claim 6, in a CA directory user scheme comprising the steps of:

the CA executing:

(i) updating said search tree so as to obtain updated nodes;

(ii) computing an authentication path as induced by said updated nodes; and

(iii) authenticating at least said root modified node by a digital signature;

(iv) transmitting modified parameters to said directory;

the directory executing:

(i) applying said modification parameters, so as to obtain authenticated directory root value;

(ii) verifying that the authenticated CA root value matched the authenticated directory value.

9. A method according to Claim 6, in a CA user scheme comprising:

the CA executing:

(i) updating said search tree so as to obtain updated nodes;

(ii) computing an authentication path as induced by said updated nodes; and

(iii) authenticating at least said root modified node by a digital signature;

(v) transmitting induced sub-tree to said user;

the user executing:

(iii) intersecting said induced sub-tree with user self path and obtaining user authenticated root value;

(iv) verifying that the authenticated CA root value matched the authenticated user value.

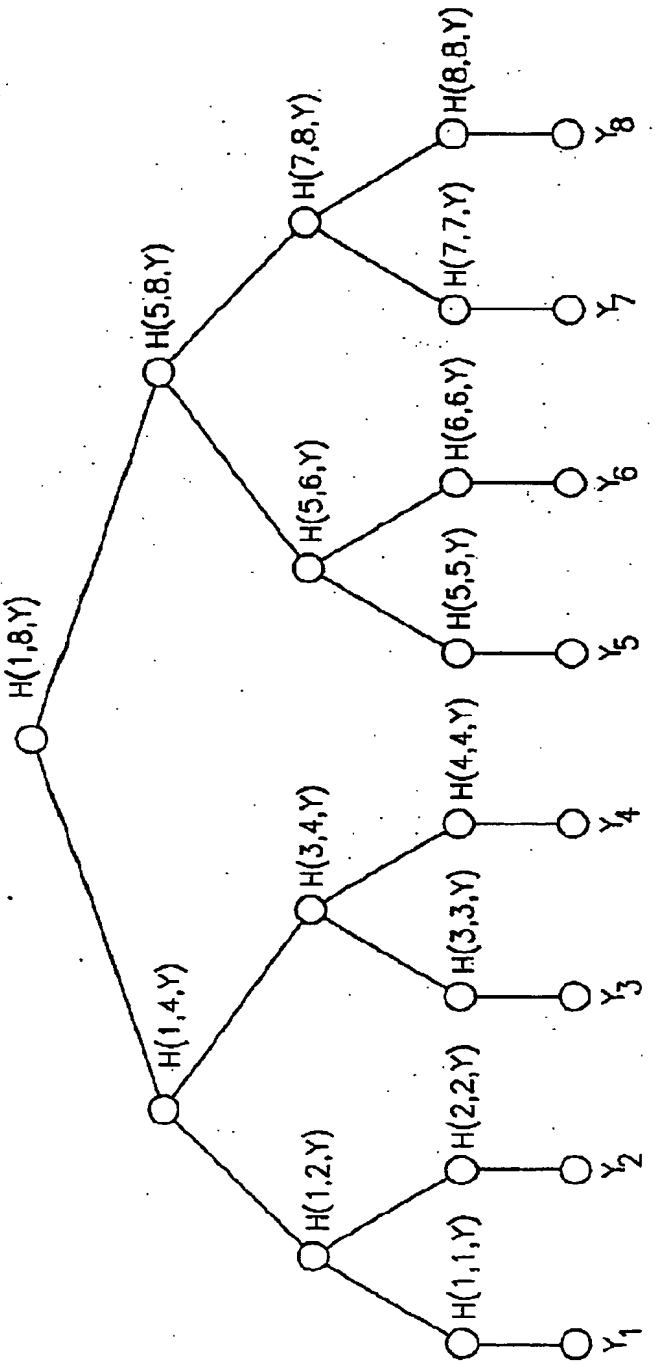

H(1,8,Y)
H(5,8,Y)
H(1,4,Y)
H(7,8,Y)
H(5,6,Y)
H(3,4,Y)
H(1,2,Y)
H(8,8,Y)
H(7,7,Y)
H(6,6,Y)
H(5,5,Y)
H(4,4,Y)
H(3,3,Y)
H(2,2,Y)
H(1,1,Y)
Y8
Y7
Y6
Y5
Y4
Y3
Y2
Y1

FIG. 1

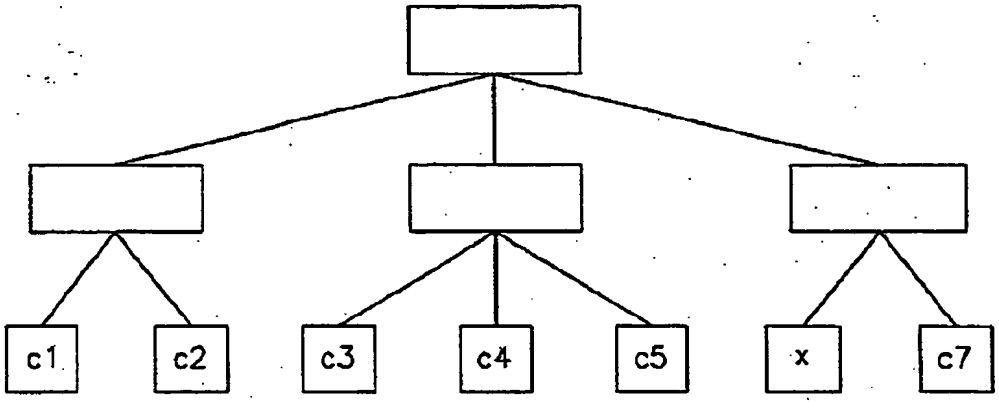


FIG.2A

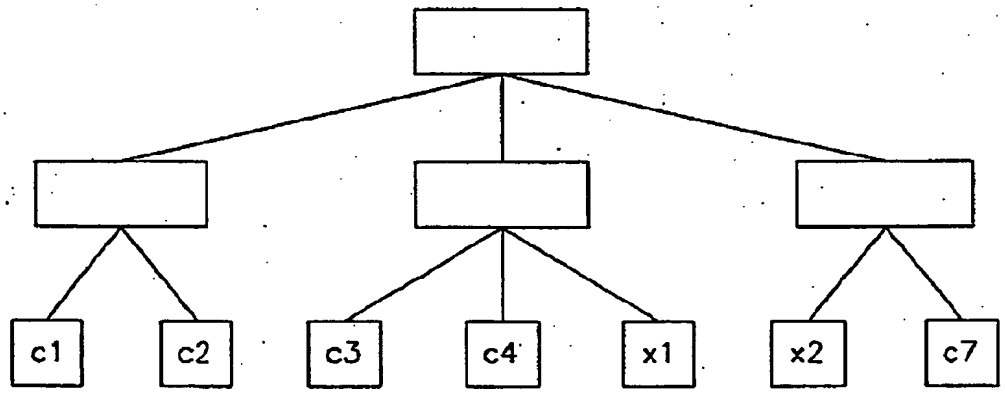


FIG.2B

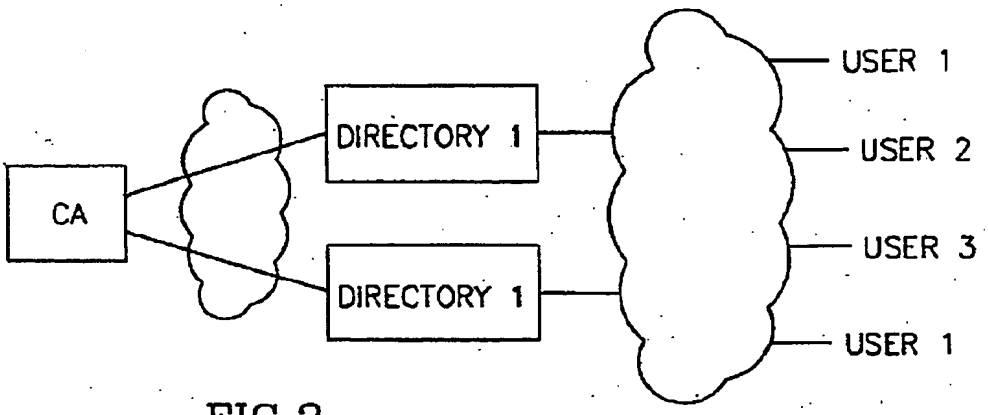


FIG.3

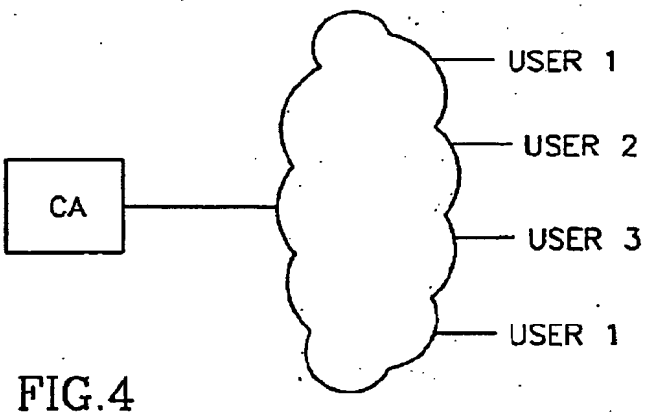


FIG.4

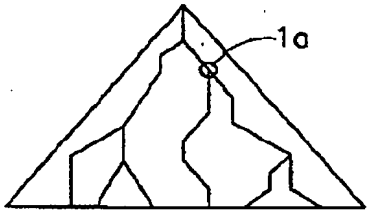


FIG.5A

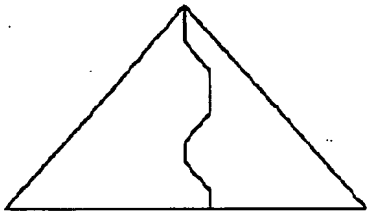


FIG.5B

ABSTRACT

1 Abstract

A memory containing an authenticated search tree that serves for authenticating membership or non membership of items in a set. The authenticated search tree including a search tree having nodes and leaves and being associated with a search scheme. The nodes including dynamic search values and the leaves including items of the set. The nodes are associated, each, with a cryptographic hash function value that is produced by applying a cryptographic hash function to the cryptographic hash values of the children nodes and to the dynamic search value of the node. The root node of the authenticated search tree is authenticated by a digital signature.